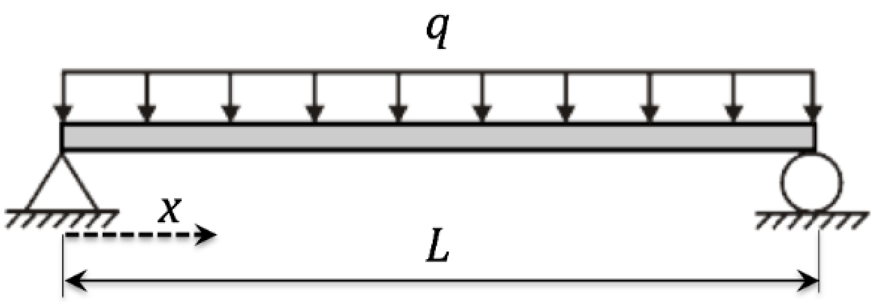
# Project 02 - Engineering Computation

**Sydney Loats (SML4267)**

## Section I: Purpose and Motivation




The relevant governing equation for this beam is given by:

$$\mathrm{EI}\,\frac{\partial^2 y}{\partial x^2} = -\frac{q}{2}(x^2 - xL)$$

The purpose of this project is to develop and implement a numeric solver for the above beam using C++. The equation that we are solving for will represent the displacement of the beam at a certain value of x on the beam based on the forces and moments that are acting on the beam. I developed this solver using the Matrix and Vector classes we have been working on all semester, as well as numerical methods to solve matrices, such Gauss-Seidel. This project is interesting to me because it combines knowledge that I am learning in multiple classes to accomplish this numeric solver.

## Section II: Preliminary Work

The preliminary work consists of Part A and Part B. Part A is shown in Figures I and II, and Part B is shown in Figure III.

<u>Figure I:</u>

(A)

$$EI\frac{d^2y}{dx^2} = -\frac{q}{2}(x^2 - xL) \qquad y(x=0)=0, \quad y(x=L)=0$$

$$EI\int\frac{d^2y}{dx^2} = -\frac{q}{6}x^3 + \frac{q}{4}x^2L + C_1$$

$$EI\int\int\frac{d^2y}{dx^2} = -\frac{q}{24}x^4 + \frac{q}{12}x^3L + C_1x + C_2$$

$$C_2 = 0$$

$$EIy = -\frac{q}{24}x^4 + \frac{q}{12}x^3L + C_1x$$

$$0 = -\frac{q}{24}L^4 + \frac{2q}{24}L^4 + C_1L \qquad -\frac{q}{24}L^4 = C_1L \qquad C_1 = -\frac{q}{24}L^3$$

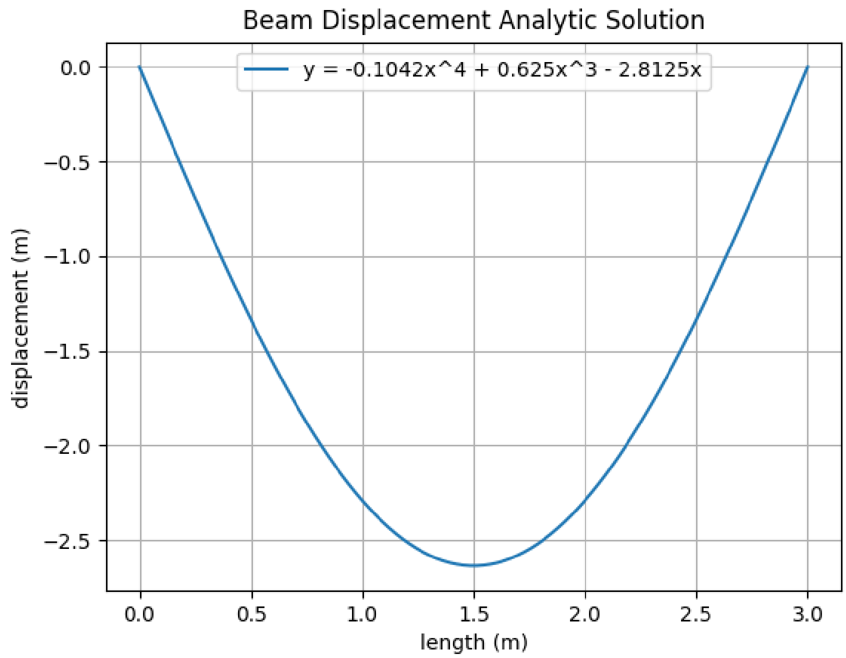
$$\boxed{y = \frac{1}{EI}\left(-\frac{q}{24}x^4 + \frac{q}{12}x^3L - \frac{q}{24}xL^3\right)}$$

plugging in $L=3m$, $EI = 600\,Nm^2$, $q = 1500\,N/m$

$$\boxed{y = -0.104x^4 + 0.625x^3 - 2.8125x}$$

I first computed the analytic solution by hand by integrating the governing equation twice and using the given relationships: y(x=0) = 0 and y(x=L) = 0. The first boxed equation in Figure I is the analytic solution in terms of L (length), EI (modulus of elasticity and moment of area), and q (distributed load per unit length). The second boxed equation is the analytic solution when L = 3m, EI = 600 Nm$^2$, and q = 1500 N/m.

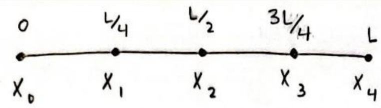
<u>Figure II:</u>




I graphed the analytic solution using the second boxed equation from Figure I. The beam displacement is parabolic where the maximum values are 0 at length=0m and length=3m. The minimum value is about -2.64m.

Figure III:

(B) $\left[0,\ \frac{L}{4},\ \frac{L}{2},\ \frac{3L}{4},\ L\right]$




for $x_1$: $EI\,\dfrac{y_2 - 2y_1 + y_0}{\Delta x^2} = -\dfrac{q}{2}(x_1^2 - x_1 L) = -\dfrac{q}{2}\left(\dfrac{L^2}{16} - \dfrac{L^2}{4}\right)$

for $x_2$: $EI\,\dfrac{y_3 - 2y_2 + y_1}{\Delta x^2} = -\dfrac{q}{2}(x_2^2 - x_2 L) = -\dfrac{q}{2}\left(\dfrac{L^2}{4} - \dfrac{L^2}{2}\right)$

for $x_3$: $EI\,\dfrac{y_4 - 2y_3 + y_2}{\Delta x^2} = -\dfrac{q}{2}(x_3^2 - x_3 L) = -\dfrac{q}{2}\left(\dfrac{9L^2}{16} - \dfrac{3L^2}{4}\right)$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 \\ -\frac{q}{2}\left(-\frac{3L^2}{16}\right) \\ -\frac{q}{2}\left(-\frac{1}{4}L^2\right) \\ -\frac{q}{2}\left(-\frac{3}{16}L^2\right) \\ 0 \end{bmatrix}$$

To visualize how to solve the matrix with 2nd order central differencing, I first discretized with 5 points spanning the domain [0, L] to get values for $x_0$, $x_1$, $x_2$, $x_3$, and $x_4$. I then discretized the governing equation generically using 2nd order central differencing to derive the three equations for $x_1$, $x_2$, and $x_3$. Given the formulas y(x=0) = 0, and y(x=L) = 0, I was able to fill the first and last rows, and with the discretized equations I was able to fill the rest of the matrix.

**Section III: Implementation**

Compiling the Code:

1. Type `g++ -o beamer beamer.cpp beam.cpp matrix.cpp vector.cpp` into the command line.

Running the Code:

1. Type `./beamer 20 3 600 1500` into the command line to run the code with a size of n = 20. You will receive an output similar to the following:

```
Beamer Solver for Analytic Solution of a Beam
--------------------------------------------
n: 20
L (m): 3
EI (Nm^2): 600
q (N/m): 1500

Number of iterations required to find the solution: 541
Elapsed wallclock time (s): 0.006863
Constant mesh interval, h (step size): 0.157895
l2 error: 0.00227957
```

2. Type ./beamer into the command line and to view how to implement correctly. The code will notify you that there is an error if you do not have the correct number of inputs. The following message shows the message you will receive:

```
error: wrong number of inputs

usage: ./beamer [n] [L] [EI] [q] <turbo>

where:
[n]           number of points
[L]           length of the beam
[EI]          (Young's modulus)*(2nd moment of intertia)
[q]           distributed load per unit length
<turbo>       optional argument (1=turbo mode)
```

**Section IV: Analysis Runs**

<u>Results (non-turbo mode):</u>

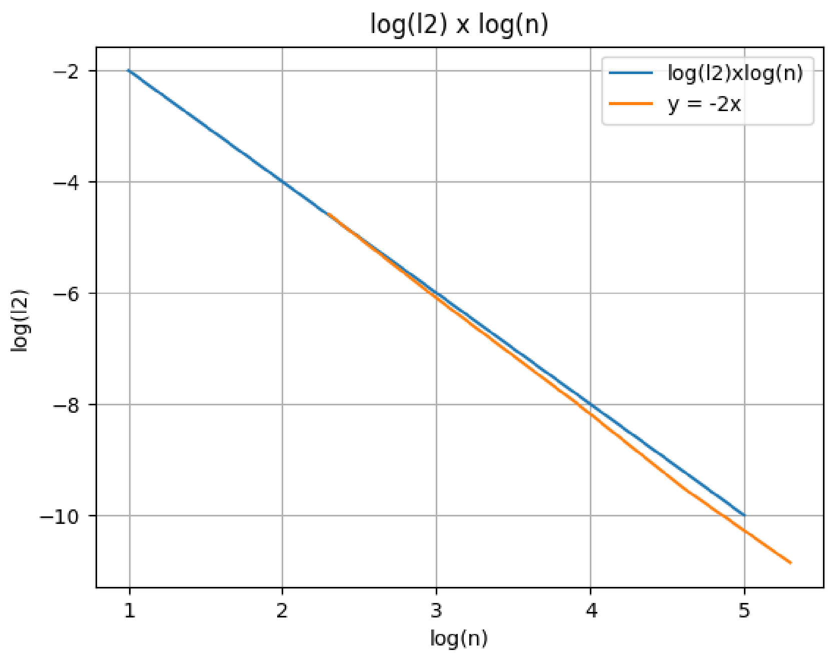
| n | h | l2 error | # of iterations | wall-clock time (secs) |
|---|---|---|---|---|
| 10 | 0.333333 | 0.0101603 | 133 | 0.001064 |
| 20 | 0.157895 | 0.00227957 | 541 | 0.008169 |
| 50 | 0.0612245 | 0.000340377 | 3144 | 0.123086 |
| 100 | 0.030303 | 7.40612E-5 | 11439 | 1.37639 |

| | | | | |
|---|---|---|---|---|
| 200 | 0.0150754 | 1.9358E-5 | 40620 | 19.4254 |

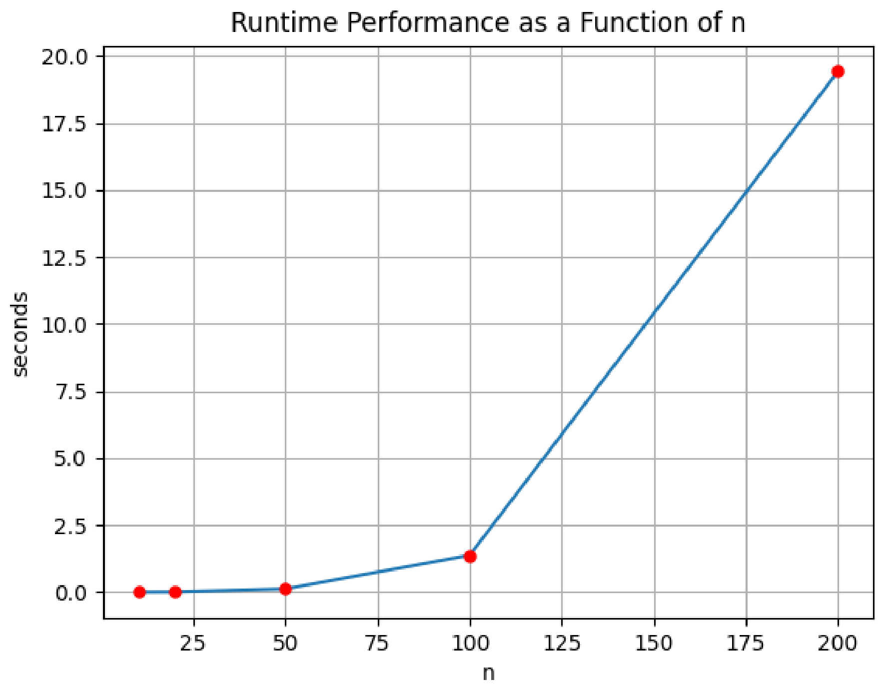
**Section V: Verification and Run Time Performance**

Figures IV and V are plots that I've created to verify my work and evaluate run time performance.

<u>Figure IV:</u>




This plot is a log/log graph of l2 errors as a function of n. Based on our knowledge of truncation error of the numerical method, the expected equation for error is y = -2x. The graph of log(l2)xlog(n) is very similar to the expected y=-2x, s these results indicate that my code is running correctly.




This plot is runtime performance as a function of n. As the value of n increases, the amount of time it takes for the code to run increases, so these results are as expected. When the values of n are smaller, the increase in the amount of time is smaller, but when the value of n increases to 200, the runtime increases by a lot. These results indicate that my code is running correctly.

**Section VI: Performance Modifications**

I made modifications to the code called "turbo mode" to speed up the time it takes to run the code for large numbers of n, such as n=200. I created another optional value that can be input at the end of the command line, where 1 = turbo mode and 0 = non turbo mode. I implemented turbo mode in the method in the Matrix class that solves the matrix using Gauss-Seidel and added methods in both the Matrix and Beam class to check if turbo mode was enabled, and to set turbo mode. When I approximate the solution in the Beam class, if turbo mode is enabled, I set that value to true in the Matrix class and solve. The way the faster code is implemented in the Matrix class is to avoid multiplying by zero many times. The nature of the matrix is shown in Figure II, where only the diagonal and surrounding values are populated with non-zero values. This means that when I am solving using Gauss-Seidel, the code is multiplying by zero many times. When turbo mode is implemented, instead of running through the whole matrix and multiplying by many zeros, the method only loops through the three values of $x_i$, $x_{i-1}$, and $x_{i+1}$. Doing this cut the time it took to solve for n=200 from 19.4254 seconds to 0.8472 seconds, making runtime almost 23x faster.

The following table includes values for running the code using turbo mode:

| n | h | l2 error | # of iterations | wall clock time turbo mode (secs) |
|---|---|---|---|---|
| 10 | 0.333333 | 0.0101603 | 133 | 0.000741 |
| 20 | 0.157895 | 0.00227957 | 541 | 0.003946 |
| 50 | 0.0612245 | 0.000340377 | 3144 | 0.036376 |
| 100 | 0.030303 | 7.40612E-5 | 11439 | 0.147541 |
| 200 | 0.0150754 | 1.9358E-5 | 40620 | 0.8472 |

**Section VII: Optional Photo**

How I feel about computational engineering. It's my major so I'm learning to love it. Thanks for a great semester Professor Schulz!

**Citations:**

Suzzene, Shaikh. “Harold Thumbs up , PNG Download - Thumbs up Meme Png, Transparent PNG - Kindpng.” *KindPNG.com*, https://www.kindpng.com/imgv/JhobJm_harold-thumbs-up-png-download-thumbs-up-meme/.